A contraction method with implementable proximal regularization for linearly constrained convex programming

Bingsheng He[†1] and **Xiaoming Yuan**[‡2]

[†]Department of Mathematics, Nanjing University, Nanjing, 210093, China
[‡]Department of Mathematics, Hong Kong Baptist University, Hong Kong, China

February 21, 2011

Abstract. The proximal point algorithm (PPA) is classical, and the resulting proximal subproblems may be as difficult as the original problem. In this paper, we show that with appropriate choices of proximal parameters, the application of PPA to the linearly constrained convex programming can result in easy proximal subproblems. In particular, under some practical assumptions on the objective function, these proximal subproblems become implementable in the sense that they all have closed-form solutions or can be efficiently solved up to a high precision. We thus present a contraction method with implementable proximal regularization for linearly constrained convex programming, and its global convergence is proved easily under the analytic framework of contraction type methods.

Keywords. Convex programming, linear constraints, proximal point algorithm, contraction method, resolvent operator

1 Introduction

In this paper, we consider the linearly constrained convex programming:

$$\min\{\theta(x)\,|\,Ax=b,\,x\in\mathcal{X}\}, \tag{1.1}$$

where $\theta(x):\Re^n\to\Re$ is a convex (not necessary smooth) function, $A\in\Re^{m\times n}$, $b\in\Re^m$ and $\mathcal{X}\subset\Re^n$ is a closed convex set. Throughout, the solution set of (1.1) is denoted by $\mathcal{X}^*$ and it is assumed to be nonempty.

Among existing methods for solving (1.1) is the classical augmented Lagrangian method (ALM) proposed originally in [13, 17]. In [19], the close relevance of ALM to the proximal point algorithm (PPA) [16] has been delineated. More specifically, it

[1] This author was supported by the NSFC grant 10971095 and Doctoral Fund of Ministry of Education of China 20060284001. Email: hebma@nju.edu.cn

[2] Corresponding author. This author was supported by a grant from Hong Kong General Research Fund. Email: xmyuan@hkbu.edu.hk

was shown in [19] that ALM for (1.1) reduces to the application of PPA to the dual of (1.1).

In this paper, we are interested in the direct application of PPA to (1.1). Let the resolvent operator of θ be defined by

$$x := \big(I + \frac{1}{r}\partial\theta\big)^{-1}(a) = \mathrm{Argmin}\{\theta(x) + \frac{r}{2}\|x - a\|^2 \,|\, x \in \mathcal{X}\}, \tag{1.2}$$

for any given $a \in \Re^n$ and $r > 0$. We present this paper under the following assumption:

Assumption: The objective function $\theta(x)$ is "simple" in the sense that its resolvent operator defined by (1.2) has a closed-form representation or it can be efficiently solved up to a high precision.

Under this assumption, we will show that the application of PPA to (1.1) with appropriate proximal parameters can result in easy proximal subproblems. Note that this assumption is not stringent for many concrete applications arising in image processing, compressive sensing, statistics, engineering, etc., see e.g. [8] and Section 4. Moreover, we will show that a contraction type method (in the sense of [1]) can be easily derived based on the output of the direct application of PPA to (1.1). Thus, a contraction type method with implementable proximal regularization is proposed for solving (1.1). As we will prove, the convergence of the new method can be easily proved under the analytic framework of contraction type methods.

The rest of the paper is organized as follows. In Section 2, we provide some preliminaries which are useful for our subsequent analysis. In Section 3, we present the new method, and prove its convergence. In Section 4, we present some concrete applications of (1.1) and elaborate on the implementation of the new method for these applications. We also report preliminary numerical results to verify the efficiency of the new method. In Section 5, we show that our discussion for (1.1) can be easily extended to a general case with linear inequalities. Finally, in Section 6, we make some conclusions.

2 Preliminaries

In this section, we first provide some notations and well known facts. Then, we derive the variational inequality reformulation of (1.1), based on which the convergence of the new algorithm will be proved.

We denote by $\partial\theta(x)$ the subdifferential of the convex function $\theta(x)$:

$$\partial\theta(x) := \{d \in \Re^n \,|\, \theta(y) - \theta(x) \geq d^T(y - x), \ \ \forall\, y \in \Re^n\},$$

and each $d \in \partial\theta(x)$ is called a subgradient of $\theta(x)$, see [18]. Let $f(x) \in \partial\theta(x)$. Then, we have

$$\theta(y) \geq \theta(x) + (y - x)^T f(x), \quad \forall x, y \in \Re^n \tag{2.1}$$

and

$$(x-y)^T(f(x)-f(y)) \geq 0, \quad \forall x, y \in \Re^n, \tag{2.2}$$

which implies that the mapping f is monotone.

Now, we show that (1.1) can be characterized by a variational inequality (VI), see e.g. [11]. By attaching a Lagrange multiplier vector $\lambda \in R^m$ to the linear constraint $Ax - b = 0$, the Lagrangian function of (1.1) is:

$$L(x, \lambda) = \theta(x) - \lambda^T(Ax - b),$$

which is defined on $\mathcal{X} \times \Re^m$. Then, by deriving the optimality condition of (1.1), we can easily see that (1.1) amounts to finding a pair of (x^*, λ^*) which satisfies

$$\left\{ \begin{array}{ll} x^* \in \mathcal{X}, & (x - x^*)^T\{f(x^*) - A^T\lambda^*\} \geq 0, \quad \forall\, x \in \mathcal{X}, \\ & Ax^* - b = 0, \end{array} \right. \tag{2.3}$$

where $f(x^*) \in \partial(\theta(x^*))$. By denoting

$$u = \begin{pmatrix} x \\ \lambda \end{pmatrix}, \quad F(u) = \begin{pmatrix} f(x) - A^T\lambda \\ Ax - b \end{pmatrix} \quad \text{and} \quad \Omega = \mathcal{X} \times \Re^m, \tag{2.4}$$

the system (2.3) can be characterized by the following variational inequality reformulation denoted by VI(Ω, F):

$$u^* \in \Omega, \qquad (u - u^*)^T F(u^*) \geq 0, \qquad \forall\, u \in \Omega. \tag{2.5}$$

Recall the monotonicity implied by (2.2). We thus know that VI(Ω, F) (2.5) is monotone. In addition, since the solution set of (1.1) is assumed to be nonempty, the solution set of VI(Ω, F), denoted by Ω^*, is also nonempty.

3 The new method

In this section, we present a contraction method with implementable proximal regularization for (1.1) and prove its global convergence. But, at the beginning we first introduce the motivation.

3.1 Motivation

For solving the VI reformulation (2.5), we can apply the classical proximal point algorithm (PPA) which was proposed originally in [16] and then developed concretely in [20]. More specifically, with the given iterate u^k, PPA solves the following proximal subproblem to generate the new iterate $u^{k+1} \in \Omega$:

$$(u' - u^{k+1})^T\{F(u^{k+1}) + G(u^{k+1} - u^k)\} \geq 0, \quad \forall\, u' \in \Omega, \tag{3.1}$$

where $G \in \Re^{(n+m)\times(n+m)}$ is a positive definite matrix and it plays the role of proximal regularization parameter. A particular choice of G is that $G = \beta \cdot I$ where $\beta > 0$ and I is the identity matrix, regularizing the proximal terms $u^{k+1} - u^k$ in the uniform way.

For the general case of (3.1) with a generic proximal regularization matrix G, it is also difficult to solve the proximal subproblem (3.1). In fact, (3.1) may be as difficult as the original problem (2.5). Because of this difficulty, the original PPA (3.1) merely makes theoretical senses and it does not yield implementable algorithms. This difficulty thus has inspired the rich literature on approximate PPAs, in order to solve the proximal subproblem (3.1) approximately subject to certain criteria, see, e.g. [2, 10, 12, 20].

In this paper, under the **Assumption**, we show that the exact solution of the proximal subproblem (3.1) can be easily solved provided that the proximal regularization matrix G is chosen judiciously. More specifically, we choose G as follows:

The proximal regularization matrix G is:

$$G = \begin{pmatrix} rI_n & -A^T \\ -A & sI_m \end{pmatrix}, \tag{3.2}$$

where $r > 0$ and $s > 0$ are chosen as $rs > \|A^TA\|$ in order to ensure the positive definiteness of G.

Then, the proximal subproblem (3.1) with the particular choice (3.2) of G reduces to

$$\begin{pmatrix} x - x^{k+1} \\ \lambda - \lambda^{k+1} \end{pmatrix}^T \left\{ \begin{pmatrix} f(x^{k+1}) - A^T\lambda^{k+1} \\ Ax^{k+1} - b \end{pmatrix} + \begin{pmatrix} rI_n & -A^T \\ -A & sI_m \end{pmatrix} \begin{pmatrix} x^{k+1} - x^k \\ \lambda^{k+1} - \lambda^k \end{pmatrix} \right\} \geq 0, \tag{3.3}$$

for any $(x, \lambda) \in \Omega$. Equivalently, we need to solve

$$(Ax^{k+1} - b) - A(x^{k+1} - x^k) + s(\lambda^{k+1} - \lambda^k) = 0, \tag{3.4}$$

and

$$x^{k+1} \in \mathcal{X}, \;\; (x - x^{k+1})^T\big\{f(x^{k+1}) - A^T(2\lambda^{k+1} - \lambda^k) + r(x^{k+1} - x^k)\big\} \geq 0, \;\; \forall\, x \in \mathcal{X}. \tag{3.5}$$

Note that the solution of (3.4) is given explicitly by

$$\lambda^{k+1} = \lambda^k - \frac{1}{s}(Ax^k - b) \tag{3.6}$$

and (3.5) amounts to solving the convex programming problem

$$x^{k+1} = \text{Argmin}\big\{\theta(x) + \frac{r}{2}\big\|(x - x^k) - \frac{1}{r}\,A^T(2\lambda^{k+1} - \lambda^k)\big\|^2 \,\big|\, x \in \mathcal{X}\big\}. \tag{3.7}$$

Thus, with the solved λ^{k+1} by (3.6) and the **Assumption**, the problem (3.7) is “simple” in the sense that it has a closed-form solution or can be solved up to a high precision.

Overall, with the particular choice (3.2) and the **Assumption**, the proximal subproblem (3.1) yielded by PPA for (2.5) is now implementable.

Moreover, we can easily show that the output of (3.1) provides us the possibility to construct a contraction type method for (3.1) in the sense of [1]. Recall that according to the definition of [1], if a method generates an iterative sequence $\{u^k\}$ satisfying the following property:

$$\|u^{k+1} - u^*\|_G \le \|u^k - u^*\|_G - c\|u^k - u^{k+1}\|_G, \quad \forall u^* \in \Omega^*, \tag{3.8}$$

where $c > 0$ is a constant and G is a positive definite matrix. Then, we call this method a contraction type method for (3.1). To see this fact, let $u^{k+1} = (x^{k+1}, \lambda^{k+1})$ be generated by (3.1) with the particular choice of G (3.2). First, because $u^* \in \Omega$, it follows from (3.1) that

$$(u^{k+1} - u^*)^T\{G(u^k - u^{k+1}) - F(u^{k+1})\} \ge 0.$$

On the other hand, since $u^{k+1} \in \Omega$ and u^* is a solution of VI(Ω, F), we have

$$(u^{k+1} - u^*)^T F(u^*) \ge 0.$$

Adding the last two inequalities and using the monotonicity of F, we obtain

$$(u^{k+1} - u^*)^T G(u^k - u^{k+1}) \ge (u^{k+1} - u^*)^T(F(u^{k+1}) - F(u^*)) \ge 0.$$

Consequently, we have

$$(u^k - u^*)^T G(u^k - u^{k+1}) \ge \|u^k - u^{k+1}\|_G^2, \quad \forall\, u^* \in \Omega^*, \tag{3.9}$$

which indicates that $-G(u^k - u^{k+1})$ is a descent direction of the unknown distance function $\frac{1}{2}\|u - u^*\|_G^2$ at the point u^k. Thus, if we move along the direction $-G(u^k - u^{k+1})$ from the previous point u^k by an appropriate step size, the new iterate is expected to be closer to the solution set Ω^*.

3.2 Algorithm

Based on the previous analysis, we are now ready to present the new contraction method with implementable proximal regularization for (1.1). For convenience, from now on, we denote by $\tilde{u}^k = (\tilde{x}^k, \tilde{\lambda}^k)$ the output of the proximal subproblem (3.1) with the given iterate $u^k = (x^k, \lambda^k)$.

Algorithm 1: A contraction method with implementable proximal regularization for (1.1)

Step 0. Let the **Assumption** be satisfied. Choose G according to (3.2). Let $\gamma \in (0, 2)$. Take $(x^0, \lambda^0) \in \Re^n \times \Re^m$.
Step k. $(k \geq 0)$ Let

$$\tilde{\lambda}^k = \lambda^k - \frac{1}{s}(Ax^k - b), \tag{3.10a}$$

and

$$\tilde{x}^k = \text{Argmin}\big\{\theta(x) + \frac{r}{2}\big\|(x - x^k) - \frac{1}{r}\, A^T(2\tilde{\lambda}^k - \lambda^k)\big\|^2 \,\big|\, x \in \mathcal{X}\big\}. \tag{3.10b}$$

Set

$$\left(\begin{array}{c} x^{k+1} \\ \lambda^{k+1} \end{array}\right) = \left(\begin{array}{c} x^{k} \\ \lambda^{k} \end{array}\right) - \gamma \left(\begin{array}{c} x^{k} - \tilde{x}^k \\ \lambda^{k} - \tilde{\lambda}^k \end{array}\right). \tag{3.11}$$

Remark 3.1. The reason why we restrict $\gamma \in (0, 2)$ will be clear in the next theorem of convergence.

3.3 Convergence

Now, we show the global convergence of the proposed algorithm. In fact, the global convergence of the proposed algorithm can be proved easily under the analytic framework of contraction type methods.

Theorem 3.2. *The sequence $\{u^k = (x^k, \lambda^k)\}$ generated by the proposed Algorithm 1 for* (1.1) *converges to the solution set of (1.1).*

Proof. First, with the new notation $\tilde{u}^k = (\tilde{x}^k, \tilde{y}^k)$, the inequality (3.9) can be rewritten accordingly into

$$(u^k - u^*)^T G(u^k - \tilde{u}^k) \geq \|u^k - \tilde{u}^k\|_G^2, \quad \forall\, u^* \in \Omega^*. \tag{3.12}$$

Thus, we have

$$\begin{aligned} \|u^{k+1} - u^*\|_G^2 &= \|u^k - \gamma(u^k - \tilde{u}^k) - u^*\|_G^2 \\ &= \|u^k - u^*\|_G^2 - 2\gamma(u^k - u^*)^T G(u^k - \tilde{u}^k) + \gamma^2\|u^k - \tilde{u}^k\|_G^2 \\ &\leq \|u^k - u^*\|_G^2 - \gamma(2 - \gamma)\|u^k - \tilde{u}^k\|_G^2, \end{aligned}$$

where the inequality follows from (3.12). Thus, the sequence $\{u^k = (x^k, \lambda^k)\}$ generated by the proposed algorithm is Fejér monotone with respect to the solution set Ω^*. The rest of the proof becomes standard by using the fact of the Fejér monotonicity. Thus, we omit it. □

Remark 3.3. Note that the condition $\gamma \in (0, 2)$ is to ensure that the proposed Algorithm 1 is a contraction type method, and its reason is obvious based on the proof of Theorem 3.2.

4 Applications and preliminary numerical experiments

In this section, we provide some concrete applications of (1.1) where the **Assumption** is practical. It is easy to see that the computation of each iteration of the proposed Algorithm 1 is dominated by the subproblem (3.10b). Hence, for these applications where the **Assumption** is satisfied, the proposed Algorithm 1 is of particular interest because of its easy implementation.

We report some preliminary numerical results, showing supportively that the proposed Algorithm 1 is capable of solving various concrete applications of (1.1) efficiently. All the codes were written by Matlab 2008b version and all the numerical experiments were done on a Dell desktop computer with Intel (R) Core 2 Quad CPU Q9400 with 2.66GHz and 3.5GB RAM.

4.1 Nearest correlation matrix problem

It is easy to see that our discussion for (1.1) can be easily extended to the more general setting where $\theta(x) : \Re^{l\times n} \to \Re$ is a convex (not necessary smooth) function, $A : \Re^{l\times n} \to \Re^{l\times n}$ is a linear operator, $b \in \Re^{l\times n}$ and $\mathcal{X} \subset \Re^{l\times n}$. Now, we present two applications in this general setting with matrix variables.

Let $C \in \Re^{n\times n}$ be a given symmetric matrix. The nearest correlation matrix problem launched in [14] is:

$$\min\{\frac{1}{2}\|X - C\|_F^2 \mid \mathrm{diag}(X) = e,\, X \in S_+^n\}, \tag{4.1}$$

where $e \in \Re^n$ is the vector whose entries are all 1, S_+^n denotes the cone of positive definite symmetric matrices, $\mathrm{diag}(X)$ denotes the diagonal matrix of X, and $\|\cdot\|_F$ denotes the matrix Fröbenis norm:

$$\|C\|_F = \Big(\sum_{i=1}^n \sum_{j=1}^n |C_{ij}|^2\Big)^{1/2}.$$

For (4.1), the subproblem (3.10b) is specified into:

$$\tilde{X}^k = \mathrm{Argmin}\Big\{\frac{1}{2}\|X - C\|_F^2 + \frac{r}{2}\big\|(X - X^k) - \frac{1}{r}\,(2\mathrm{diag}(\tilde{\lambda}^k) - \mathrm{diag}(\lambda^k))\big\|^2 \,\big|\, X \in S_+^n\Big\}, \tag{4.2}$$

where

$$\tilde{\lambda}^k = \lambda^k - \frac{1}{s}(\mathrm{diag}(X^k) - e).$$

Thus, the **Assumption** is satisfied. In fact, by noticing the fact that (4.2) is equivalent to

$$\min\{\|X - \tfrac{1}{1+r}(rX^k + (2\mathrm{diag}(\tilde{\lambda}^k) - \mathrm{diag}(\lambda^k)) + C)\|_F^2 \mid X \in S_+^n\},$$

it is easy to see that the closed-form solution of (4.2) is given by

$$\tilde{X}^k = P_{S_+^n}[\tfrac{1}{1+r}(rX^k + 2\mathrm{diag}(\tilde{\lambda}^k) - \mathrm{diag}(\lambda^k) + C)], \tag{4.3}$$

where $P_{S^n_+}$ denotes the projection operator onto S^n_+ which can be completed by an eigenvalue decomposition. In our experiments, we apply the standard subroutine of Matlab to realize the eigenvalue decomposition.

We generate the matrix C by the following Matlab scripts:

```
rand('state',0);  C=rand(n,n);   C=(C'+C)-ones(n,n) + eye(n);
```

In this way, the matrix C is a quasi-random symmetric matrix whose elements satisfy:

$$C_{ij} \in \begin{cases} (0,2), & \text{if} \quad i = j; \\ (-1,1), & \text{otherwise.} \end{cases}$$

To apply the proposed Algorithm 1 to solve (4.1), we note that the mapping $\operatorname{diag}(X)$ can be regarded as a projection operator. Thus, the norm of this operator is 1. Therefore, the condition on r and s to ensure the positive definiteness of the proximal regularization matrix G can be simply taken as

$$rs > 1.$$

In implementation, we take $r = 2$ and $s = 1.01/r$, and the the iteration starts with the initial iterate $X^0 = I$ and $\lambda^0 = 0$. Now, we elaborate on the stopping criterion for (4.1). By observing (2.3) and (3.3), it is easy to see that the term $G(u^{k+1} - u^k)$ measures the violation of the optimality condition for the iterate $u^{k+1} = (x^{k+1}, \lambda^{k+1})$. Therefore, we employ the following stopping criterion for solving (4.1):

$$\max\{\max_{ij} |X^k_{ij} - \tilde{X}^k_{ij}|, \ \max_j |\lambda^k_j - \tilde{\lambda}^k_j|\} \leq 10^{-5}.$$

In Table 1, we report the performance of Algorithm 1 for solving (4.1). In particular, we report the number of iteration ("No. It") and the computation time in seconds ("CPU.") for various scenarios of n and γ.

Table 1. Performance of Algorithm 1 for (4.1)

	$\gamma = 1$		$\gamma = 1.5$	
n	No. It	CPU.	No. It	CPU.
100	30	0.34	22	0.23
200	33	1.88	25	1.33
500	39	16.76	27	11.55
1000	48	151.40	31	97.91

According to the data in Table 1, we observed that the numerical performance of the proposed Algorithm 1 is sensitive to the value of the parameter γ. In particular, for (4.1), the performance of Algorithm 1 with $\gamma = 1.5$ is faster than the case with $\gamma = 1$ by about 35%. Recall that the proposed Algorithm 1 with $\gamma = 1$ reduces to the classical PPA with the proximal regularization matrix G defined in (3.2). Thus, these data in Table 1 validate the necessity of the additional step (3.11) of the proposed Algorithm 1. In fact, based on our numerical experiences, we recommend to take $\gamma \in [1.2, 1.9]$ empirically even though we have proved that $\gamma \in (0, 2)$ is valid theoretically.

4.2 Matrix completion problem

The matrix completion problem is to complete a unknown matrix from a small number of entries, and it captures many applications in such areas as statistics, machine learning, system identification and computer vision. We refer to, e.g. [7], for some remarkable results on this topic. More specifically, let $M \in \Re^{l\times n}$ be a given matrix whose rank is usually low and Π denote a subset of the indices of entries of a matrix which indicates the sample data. The mathematical form of the matrix completion problem is

$$\min\{\|X\|_* \mid X_\Pi = M_\Pi\}, \tag{4.4}$$

where $\|X\|_*$ denotes the nuclear norm which is defined as the sum of all singular values of X, and $(\cdot)_\Pi : \Re^{l\times n} \to \Re^{l\times n}$ is the sampling operator defined by $X_{ij} = M_{ij}$ if $(i,j) \in \Pi$ and 0 otherwise. Obviously, the sampling operator is linear, and it is also a projection operator. Thus the condition on r and s in the implementation of the proposed Algorithm 1 is also easy: simply let $rs > 1$.

Now, we look at the subproblem (3.10b) when the proposed Algorithm 1 is applied to solve (4.4). In this case, (3.10b) is specified into

$$\tilde{X}^k \quad = \quad \text{Argmin}\Big\{\frac{1}{r}\|X\|_* + \frac{1}{2}\big\|(X - X^k) - \frac{1}{r}(2\tilde{Z}^k_\Pi - Z^k_\Pi)\big\|^2_F\Big\}, \tag{4.5}$$

where

$$\tilde{Z}^k_\Pi = Z^k_\Pi - \frac{1}{s}(X^k_\Pi - M_\Pi).$$

Then, the **Assumption** is satisfied because that the closed-form solution of (4.5) exists. To see it, let

$$X^k + \frac{1}{r}(2\tilde{Z}^k_\Pi - Z^k_\Pi) = U^k\Lambda^k(V^k)^T$$

be the singular value decomposition of the matrix $X^k + \frac{1}{r}(2\tilde{Z}^k_\Pi - Z^k_\Pi)$. In particular, the diagonal matrix $\Lambda^k = \text{diag}(\lambda_1, \lambda_2, \cdots, \lambda_n)$ where $\lambda_i's$ are singular values of the matrix $X^k + \frac{1}{r}(2\tilde{Z}^k_\Pi - Z^k_\Pi)$. Then, the closed-form solution of (4.5) is given by:

$$\tilde{X}^k = U^k\tilde{\Lambda}^k(V^k)^T.$$

where $\tilde{\Lambda}^k = \text{diag}(\tilde{\lambda}_1, \tilde{\lambda}_2, \cdots, \tilde{\lambda}_n)$ and

$$\tilde{\lambda}_j = \max(\lambda_j - \frac{1}{r}, 0).$$

In our experiments, as in [4], we apply the well-known PROPACK package [15] to accomplish the singular value decomposition.

We form the test examples of (4.4) in the way suggested by [3]. In particular, the $n \times n$ matrix M is set as $M = M_L M_R^T$, where M_L and M_R are two $n \times r_a$ matrices sampled independently, each having i.i.d Gaussian entries. The matrix M is thus with rank r_a. The set of observed entries Π is sampled uniformly at random among all sets of cardinality of m. As explained in [3], an $n \times n$ matrix of rank r_a depends upon $d_{r_a} := r_a(2n - r_a)$ degrees of freedom. Then m/d_{r_a} is the ratio between the number of sampled entries and the 'true dimensionality' of an $n \times n$ matrix of rank r_a.

To apply the proposed Algorithm 1 for (4.4), we take $\gamma = 1.5$ and set the parameters $r = 0.005$ and $s = 1.01/r$ in (3.2). As [3], we use the stopping criterion

$$\text{relative error} = \frac{\|X^k_\Pi - M_\Pi\|_F}{\|M_\Pi\|_F} \le 10^{-4} \tag{4.6}$$

as the stopping criterion.

In Table 2, we report the numerical performance of the proposed Algorithm 1 for solving (4.4). In particular, for various scenarios of r_a, m/d_{r_a} and m/n^2, we report the number of iterations ("No. It"), computation time in seconds ("CPU."), the relative error ("RelErr") defined in (4.6). In addition, we report the violation of optimality condition ("Opti-Violation") when the stopping criterion (4.6) is satisfied. Note that due to (3.3), the violation of optimality condition can be measured by:

$$\max\{r \max_{ij} |X^k_{ij} - \tilde{X}^k_{ij}|, \ \max_{ij} |Z^k_{ij} - \tilde{Z}^k_{ij}|\}.$$

Table 2. Numerical performance of Algorithm 1 for (4.4)

Unknown matrix M				Algorithm 1			
size($n \times n$)	rankr_a	m/d_R	m/n^2	No. It	CPU.	RelErr	Opti-Violation
1000×1000	10	6	0.12	76	30.99	9.30E-5	9.19E-6
1000×1000	50	4	0.39	36	40.25	1.29E-4	2.08E-5
1000×1000	100	3	0.58	30	42.45	1.50E-4	3.82E-5

The data in Table 2 further verify the efficiency of the proposed Algorithm 1.

Note that the model (4.1) under consideration is a broad model which captures many concrete applications, and the new method is presented to solve the generic case of (4.1). Thus, the proposed method is applicable to a broad spectrum of applications. In addition to the wide applicability, our preliminary numerical results show affirmatively that the proposed method could be very efficient for a specific application of (4.1).

5 Extension

In this section, we show that the proposed results can be easily extended to the extended case of (1.1) with linear inequalities:

$$\min\{\theta(x) \,|\, Ax \ge b, \, x \in \mathcal{X}\}. \tag{5.1}$$

Similarly, by attaching a Lagrange multiplier $\lambda \in R^m_+$ to the linear inequalities $Ax \ge b$, the Lagrangian function of (5.1) is

$$L(x, \lambda) = \theta(x) - \lambda^T(Ax - b),$$

which is defined on $\mathcal{X} \times \Re^m_+$. Then, by deriving the optimality condition of (5.1), we can easily see that (5.1) amounts to finding a pair of (x^*, λ^*) which satisfies

$$\left\{ \begin{array}{ll} x^* \in \mathcal{X}, & (x - x^*)^T \{f(x^*) - A^T \lambda^*\} \geq 0, \quad \forall\, x \in \mathcal{X}, \\ \lambda^* \geq 0 & (\lambda - \lambda^*)^T (Ax^* - b) \geq 0, \qquad\qquad \forall\, \lambda \in \Re^m_+, \end{array} \right. \tag{5.2}$$

where $f(x^*) \in \partial(\theta(x^*))$. By denoting

$$u = \begin{pmatrix} x \\ \lambda \end{pmatrix}, \quad F(u) = \begin{pmatrix} f(x) - A^T \lambda \\ Ax - b \end{pmatrix} \quad \text{and} \quad \Omega = \mathcal{X} \times \Re^m_+, \tag{5.3}$$

the system (5.2) can be characterized by the following VI reformulation:

$$u^* \in \Omega, \qquad (u - u^*)^T F(u^*) \geq 0, \qquad \forall\, u \in \Omega. \tag{5.4}$$

Analogous to the analysis in Section 3, by choosing the proximal regularization matrix G in the form of (3.2), at the $(k+1)$-th iteration the proximal subproblem of the application of PPA for (5.4) is:

$$\begin{pmatrix} x - x^{k+1} \\ \lambda - \lambda^{k+1} \end{pmatrix}^T \left\{ \begin{pmatrix} f(x^{k+1}) - A^T \lambda^{k+1} \\ Ax^{k+1} - b \end{pmatrix} + \begin{pmatrix} rI_n & -A^T \\ -A & sI_m \end{pmatrix} \begin{pmatrix} x^{k+1} - x^k \\ \lambda^{k+1} - \lambda^k \end{pmatrix} \right\} \geq 0, \tag{5.5}$$

for any $(x, \lambda) \in \Omega$. Therefore, we need to solve

$$\lambda^{k+1} \geq 0, \;\; (\lambda - \lambda^{k+1})^T \{(Ax^{k+1} - b) - A(x^{k+1} - x^k) + s(\lambda^{k+1} - \lambda^k)\} \geq 0, \;\; \forall\, \lambda \in \Re^m_+, \tag{5.6}$$

and

$$x^{k+1} \in \mathcal{X}, \;\; (x - x^{k+1})^T \big\{ f(x^{k+1}) - A^T(2\lambda^{k+1} - \lambda^k) + r(x^{k+1} - x^k) \big\} \geq 0, \;\; \forall\, x \in \mathcal{X}. \tag{5.7}$$

Note that the VI (5.6) is equivalent to

$$\lambda^{k+1} \geq 0, \;\; (\lambda - \lambda^{k+1})^T \{ \frac{1}{s}(Ax^k - b) + (\lambda^{k+1} - \lambda^k)\} \geq 0, \;\; \forall\, \lambda \in \Re^m_+,$$

whose closed-form solution is given by

$$\lambda^{k+1} = P_{\Re^m_+}[\lambda^k - \frac{1}{s}(Ax^k - b)], \tag{5.8}$$

where $P_{\Re^m_+}$ denotes the projection operator onto $\Re^m_+$ under the Euclidean distance.

On the other hand, the VI (5.7) amounts to solving the convex programming problem

$$x^{k+1} = \text{Argmin}\big\{ \theta(x) + \frac{r}{2} \big\| (x - x^k) - \frac{1}{r} A^T (2\lambda^{k+1} - \lambda^k) \big\|^2 \,\big|\, x \in \mathcal{X} \big\}. \tag{5.9}$$

Thus, with the solved λ^{k+1} by (5.8) and the **Assumption**, the problem (5.9) is "simple" in the sense that it has a closed-form solution or can be solved up to a high precision.

Therefore, with the particular choice (3.2) and the **Assumption**, the application of PPA for (5.4) is also implementable in the sense that all the resulting proximal subproblems are easy. For completeness, we present a contraction method with implementable proximal regularization for (5.1) as follows, and its convergence analysis is omitted because of its similarity to Theorem 3.2.

Algorithm: A contraction method with implementable proximal regularization for (5.1)

Step 0. Let the **Assumption** be satisfied. Choose G according to (3.2). Let $\gamma \in (0, 2)$. Take $(x^0, \lambda^0) \in \Re^n \times \Re^m$.

Step k. $(k \geq 0)$ Let

$$\tilde{\lambda}^k = P_{\Re^m_+}[\lambda^k - \frac{1}{s}(Ax^k - b)], \tag{5.10a}$$

and

$$\tilde{x}^k = \text{Argmin}\big\{\theta(x) + \frac{r}{2}\big\|(x - x^k) - \frac{1}{r}\,A^T(2\tilde{\lambda}^k - \lambda^k)\big\|^2 \,\big|\, x \in \mathcal{X}\big\}. \tag{5.10b}$$

Set

$$\left(\begin{array}{c} x^{k+1} \\ \lambda^{k+1} \end{array}\right) = \left(\begin{array}{c} x^k \\ \lambda^k \end{array}\right) - \gamma \left(\begin{array}{c} x^k - \tilde{x}^k \\ \lambda^k - \tilde{\lambda}^k \end{array}\right). \tag{5.11}$$

Recall that we denote by $\tilde{u}^k = (\tilde{x}^k, \tilde{\lambda}^k)$ the output of the proximal subproblem (5.5) with the given iterate $u^k = (x^k, \lambda^k)$.

6 Conclusions

In this paper, we show that when the classical proximal point algorithm (PPA) is applied to solve some linearly constrained convex programming problems, the resulting proximal subproblems can be very easy (e.g. closed-form solutions are available) if appropriate proximal parameters are chosen judiciously. The difficulty of implementation of PPA thus can be eliminated for a broad class of problems which capture many applications. Based on this fact, a contraction method with implementable proximal regularization are proposed accordingly. This paper is an illustration of the facts that some classical methods are at the root of many efficient algorithms in the literature, and that structure-exploiting efficient algorithms could be developed by adopting classical methods with particular considerations of problems' structures we encounter.

References

[1] E. Blum and W. Oettli, Mathematische Optimierung, Econometrics and Operations Research XX, Springer Verlag, 1975.

[2] J. V. Burke and M. J. Qian, A variable metric proximal point algorithm for monotone operators, SIAM J. Con. Optimi., 37 (1998), pp. 353-375.

[3] E. J. Candès and B. Recht. Exact matrix completion via convex optimization, Foudations of Computational Mathematics, 9 (2008), pp. 717-772.

[4] J. F. Cai, Candès and Z. W. Shen, A singular value thresholding algorithm for matrix completion, SIAM J. Optim., 20 (2010), pp. 1956C1982.

[5] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Inform. Theory, 52(2)(2006), pp. 489-509.

[6] E. J. Candès and T. Tao, Near-optimal signal recovery from random projections and universal encoding strategies. IEEE Trans. Inform. Theory, 52(12) (2006), pp. 5406-5425.

[7] E. J. Candès and T. Tao, The power of convex relaxation: Near-optimial matrix completion, IEEE Trans. Inform. Theory, 56(5) (2009), pp. 2053-2080.

[8] A. Chambolle and T. Pock, A first-order primal-dual algorithms for convex problem with applications to imaging, manuscript, 2010.

[9] S. S. Chen, D. L. Donoho and M. A. Saunders. Atomic decomposition by basis pursuit. SIAM J. Sci. Comput., 20 (1999), pp. 33-61.

[10] J. Eckstein, Approximate iterations in Bregman-function-based proximal algorithms, Math. Program., 83 (1998), pp. 113-123.

[11] F. Facchinei and J.S. Pang, Finite-dimensional variational inequalities and complementarity problems, Springer, New York, 2003.

[12] O. Güler, On the convergence of the proximal point algorithm for convex minimization, SIAM J. Control Optim., 29(2)(1991), pp. 403-419.

[13] M. R. Hestenes, Multiplier and gradient methods, J. Optimi. Theory Appli., 4 (1969), pp. 303-320.

[14] N.J. Higham, Computing the nearest correlation matrix–a problem from finance, IMA J. Numer. Analysis 22 (2002), pp. 329-343.

[15] R. M. Larsen, PropackCsoftware for large and sparse svd calculation, http://sun.stanford.edu/ rmunk/PROPACK/, 2005.

[16] B. Martinet, Regularisation, d'inéquations variationelles par approximations succesives, Rev. Francaise d'Inform. Recherche Oper., 4 (1970), pp. 154-159.

[17] M. J. D. Powell, A method for nonlinear constraints in minimization problems, In Optimization edited by R. Fletcher, pp. 283-298, Academic Press, New York, 1969.

[18] R. T. Rockafellar, Convex Analysis, Princeton University Press, Princeton, NJ, 1970.

[19] R.T. Rockafellar, Augmented Lagrangians and applications of the proximal point algorithm in convex programming, Math. Oper. Res. 1 (1976), pp. 97-116.

[20] R.T. Rockafellar, Monotone operators and the proximal point algorithm, SIAM J. Control Optim., 14 (1976), pp. 877-898.